1. Conjunto de reglas, conjunto de símbolos terminales, conjuntos no terminales, símbolo inicial.
2. Metalenguaje: Es un lenguaje utilizado para describir, definir o analizar otro lenguaje, llamado el lenguaje objeto. En otras palabras, un metalenguaje es un lenguaje sobre un lenguaje. Se utiliza en la teoría de lenguajes formales, en la definición de lenguajes de programación y en la discusión de la sintaxis y la semántica de los lenguajes.

Metasímbolo: Es un símbolo que se utiliza en un metalenguaje para representar o referirse a los símbolos en el lenguaje objeto. Los metasímbolos son esenciales para definir la estructura y las reglas de un lenguaje.

BNF (Backus-Naur Form): Es una notación metalingüística, es decir, un tipo de metalenguaje, que se utiliza para representar formalmente la sintaxis de los lenguajes de programación. BNF utiliza metasímbolos como ::= (que significa “se define como”) y | (que significa “o”) para definir las reglas de producción que describen cómo se forman las oraciones válidas en un lenguaje.

Sintaxis: Se refiere al conjunto de reglas que definen cómo se pueden formar las oraciones o expresiones válidas en un lenguaje. En el contexto de los lenguajes de programación, la sintaxis define cómo se deben escribir y estructurar los programas para que sean correctamente interpretados por el compilador o el intérprete.

1. A) +123.456, 678.910, -123.456 B) BNF extendida

VT = {+,-,.}

VN = {digito}

S = +|-

P = <digito> {<digito>} [ . {<digito>} ]

1. <IfElse> ::= if <condition> : <Instructions> [ else : <Instructions> ]

<ForCycle> ::= for (<Initial value> “;” <Condition> “;” <Increase> | <Decrease>)

1. ***Análisis léxico:*** Divide el código en componentes llamados “tokens”. Cada parte del código le corresponde un token.

***Análisis sintáctico:*** Se genera un árbol de análisis sintáctico del programa una vez verificada la sintaxis del código fuente.

***Análisis semántico:*** Se realiza para verificar la coherencia del programa. Busca errores más allá de la sintaxis del programa, como en los tipos de datos, declaración de funciones, y resolución de nombres.

***Generación de código intermedio:*** Es una representación de bajo nivel del programa, cercano al lenguaje máquina, pero independiente del hardware.

***Optimización del código:*** El código intermedio se somete a técnicas para aumentar el rendimiento final del programa, como por ejemplo, la eliminación de datos redundantes, reorden de instrucciones y el uso eficiente de los registros del CPU.

***Generación de código objeto:*** El compilador traduce el código intermedio optimizado en código específico de la plataforma de destino.

1. Es una técnica que introduce redundancia al código con el propósito de evitar corrupción de datos. Se cuentan la cantidad de bits “1” que se van a enviar. Si la cantidad es impar, se añade un bit “1” extra. Si ya es par, se añade un bit “0” al código. Se debe verificar que la cantidad de bits “1” sea par para tomar la decisión de rechazar o no los datos.